REMARKS

Claims 1-82 are now pending. No claims stand allowed.

Claims 1, 13, 17, 20, 21 and 26-30 have been amended to further particularly point out and distinctly claim subject matter regarded as the invention. The text of claims 2-12, 14-16, 18-19, 22-24, is unchanged, but their meaning is changed because they depend from amended claims 1, 13, 17 and 21, respectively.

New claims 31-82 have been added by this amendment and also particularly point out and distinctly claim subject matter regarded as the invention. Claims 31-56 are In re Beauregard[1] claims corresponding to method claims 1-26. Claims 57-82 are means-plus-function claims corresponding to method claims 1-26.

No "new matter" has been added by the amendment.

The 35 U.S.C. § 112, Second Paragraph Rejection

Claim 21 stands rejected under 35 U.S.C. § 112, second paragraph, as being allegedly indefinite for failing to particularly point out and distinctly claim the subject matter applicant regards as the invention.[2] With this Amendment, claim 21 has been amended to provide proper antecedent basis. Accordingly, withdrawal of the 35 U.S.C. § 112, second paragraph rejection is respectfully requested.

---

[1] *In re Beauregard*, 53 F.3rd 1853, USPQ2nd 1383 (Fed. Cir. 1995).
[2] Office Action dated September 30, 2002 at ¶ 3.

The 35 U.S.C. § 102 Rejection

Claims 1-12 and 27 stand rejected under 35 U.S.C. § 102(b) as being allegedly

anticipated by Yellin et al.[3][4] This rejection is respectfully traversed.

"A claim is anticipated only if each and every element as set forth in the claim is found,

either expressly or inherently described, in a single prior art reference."[5] "The identical

invention must be shown in as complete detail as is contained in the ... claim."[6]

Claim 1

Claim 1 as amended recites:

A computer implemented process for managing exceptions throwable during execution of
    methods in one or more classes on a resource-constrained device, each method
    including an exception handler array defining exception handlers associated with the
    method, the process comprising:
combining the exception handler arrays for two or more methods into a single exception
    handler table.

The Examiner states:

As to claim 1, Yellin discloses managing exceptions throwable during execution of
methods in one or more classes by a machine (line 66 column 1 to line 28 column 2),
each method (each method, line 13 column 3) including an exception handler array
defining exception handlers associated with the method (the code for the exception
handlers, line 14-15 column 3), combining the exception handler arrays for two or more
methods into a single exception handler table (one table of exception handlers for all the
methods in a class, lines 16-18 column 3).[7]

---

[3] U.S. Patent No. 5,761,513.
[4] Office Action at ¶ 4.
[5] *Verdegaal Bros. v. Union Oil Co. of California*, 814 F.2d 628, 631, 2 USPQ2d 1051, 1053 (Fed. Cir. 1987).
[6] *Richardson v. Suzuki Motor Co.*, 869 F.2d 1226, 1236, 9 USPQ2d 1913, 1920 (Fed. Cir. 1989). See also,
M.P.E.P. § 2131.

With this Amendment, claim 1 has been modified to specify execution on a resource-constrained device. The resource-constrained device limitation is not disclosed in the Yellin et al. reference, so the rejection is unsupported by the art and should be withdrawn.

Dependent Claims 2-12

Claims 2-12 depend from claim 1 and thus include the "resource-constrained device" limitation. The base claim being allowable, the dependent claims must also be allowable.

Claim 3

Claim 3 recites:

> The process of claim 1 including combining all exception handler arrays for all methods in all classes in the single exception handler table.

The Examiner states:

> As to claim 3, Yellin further teaches combining all exception handler arrays for all methods in all classes in the single exception handler table (lines 19-40 column 3).[9]

The Applicants respectfully disagree. Contrary to the Examiner's statement, Yellin et al. does not disclose combining *all* exception handlers for *all* methods in all classes in the *single* exception handler table. In support of the Examiner's contention, the Examiner cites portions of Yellin et al. that speak generally about the logical relationship of exceptions. More specifically, Yellin et al. discloses organizing exceptions into a class hierarchy. Nowhere does Yellin et al. indicate exception handler arrays for *all* methods in *all* classes are combined into a *single*

---

[7] Office Action at ¶ 4.
[8] *Id.*
[9] *Id.*

24

exception handler table. Since this element is not disclosed in the <u>Yellin et al.</u> reference, the

rejection is unsupported by the art and should be withdrawn.

<u>Claim 4</u>

Claim 4 recites:

The process of claim 1 including combining all exception handler arrays for all methods in a Java package in the single exception handler table.

The Examiner states:

As to claim 4, Yellin further teaches combining all exception handler arrays for all methods in a Java package (the methods in a Java class file, line 20 column 3 and Fig. 4) in the single exception handler table (one table of exception handlers for all the methods in a class, lines 16-18 column 3).[10]

The Applicants respectfully disagree. Contrary to the Examiner's statement, <u>Yellin et al.</u>

does not disclose combining *all* exception handler arrays for *all* methods in a *Java package* in

the *single* exception handler table. In support of the Examiner's contention, the Examiner cites

portions of <u>Yellin et al.</u> that disclose a single exception handler table for methods in a single

class. Nowhere does <u>Yellin et al.</u> indicate exception handler arrays for *all* methods in a *Java*

package are combined into a *single* exception handler table. Since this element is not disclosed

in the <u>Yellin et al.</u> reference, the rejection is unsupported by the art and should be withdrawn.

<u>Claim 6</u>

Claim 6 recites:

---

[10] *Id.*
[11] *Id.*

> The process of claim 1 further including searching the exception handler table when an exception is thrown while executing one of the methods including locating a first matching exception in the single exception handler table.

The Examiner states:

> As to claim 6, Yellin further teaches searching the exception handler table (found in a tree search, lines 42-44 column 3) when an exception is thrown (an exception is thrown, line 41 column 3) while executing one of the methods (while executing the protected code block, line 65 column 3) including locating a first matching exception in the single exception handler table (the enclosing exception handlers that is applicable to the thrown exception, lines 42-44 column 3).[12]

The Applicants respectfully disagree. Contrary to the Examiner's statement, <u>Yellin et al.</u> does not disclose searching the exception handler table when an exception is thrown while executing one of the methods including locating a first matching exception in the *single* exception handler table. In support of the Examiner's contention, the Examiner cites portions of <u>Yellin et al.</u> that speak generally about the logical relationship of exceptions. More specifically, <u>Yellin et al.</u> discloses performing a search on a hierarchy representing exceptions. Nowhere does <u>Yellin et al.</u> disclose searching the exception handler table when an exception is thrown while executing one of the methods including locating a first matching exception in the *single* exception handler table. Since this element is not disclosed in the <u>Yellin et al.</u> reference, the rejection is unsupported by the art and should be withdrawn.

<u>Claim 7</u>

Claim 7 recites:

> The process of claim 6 where the searching step includes retrieving in order exception handler entries from the exception handler table and checking the type and range of each exception handler for the first matching exception handler.

The Examiner states:

> As to claim 7, Yellin further teaches retrieving in order exception handler entries (first exception handler found, line 42 column 3) from the exception handler table and checking the type and range of each exception handler (class rank is determined by position, lines 49-50 column 3) for the first matching exception handler.[13]

The Applicants respectfully disagree. Contrary to the Examiner's statement, Yellin et al. does not disclose retrieving in order exception handler entries from the exception handler table and checking the type and range of each exception handler for the first matching exception handler. Yellin et al. discloses:

> When an exception is thrown, the Java virtual machine executes the first exception handler found in a tree search of the enclosing exception handlers that is applicable to the thrown exception. To ensure that the lowest class enclosing exception handler available to the thrown exception is used, the authors of Java programs (i.e. methods) will generally order the exception handlers within each method so that the lower class exception handlers are positioned before the higher class exception handlers (where class rank is determined by position of the exception hierarchy shown in FIGS. 2 and 3).[14]

Thus, the position of an exception handler in a table is determined by the position of the exception handler in the exception hierarchy. The "checking" referred to by the Examiner is part of the process used to *order* exception handlers in an exception handler table; it is not part of the process of searching an ordered exception handler table. Since this element is not disclosed in the Yellin et al. reference, the rejection is unsupported by the art and should be withdrawn.

Claim 8

Claim 8 recites:

---

[12] *Id.*
[13] *Id.*
[14] Yellin et al. at col. 3 lines 41-50.

27

The process of claim 7 further comprising stopping searching if a current exception handler does not match and is the last handler for the top most level of protected code in an associated method.

The Examiner states:

As to claim 8, Yellin further teaches stopping searching if a current exception handler does not match and is the last handler for the top most level of protected code in an associated method (lines 19-47 column 7).[15]

The Applicants respectfully disagree. Contrary to the Examiner's statement, Yellin et al. does not disclose stopping searching if a current exception handler does not match and is the last handler for the top most level of protected code in an associated method. In support of the Examiner's contention, the Examiner cites portions of Yellin et al. that speak generally about enforcement of a policy that all normal exceptions that are throwable by an invoked method be handled by the exception handlers.[16] More specifically, Yellin et al. discloses:

...whenever a method is *invoked*, the list of normal exceptions that are declared by the called method are compared with the set of enclosing exception handlers that are applicable to the called method as a whole (step 280) ... if the invoked method's header declares any normal exceptions for which the invoking (i.e. calling) method had not established applicable exception handlers (step 282), then *execution of the invoked method is blocked* (step 284).[17]

The enforcement policy referred to by the Examiner is performed upon method *invocation*, whereas the searching claimed in claim 8 is performed when an exception is *thrown*. Since this element is not disclosed in the Yellin et al. reference, the rejection is unsupported by the art and should be withdrawn.

---

[15] Office Action at ¶ 4.
[16] Yellin et al. at col. 7 lines 8-18.
[17] *Id.* at col. 7 lines 20-41 (emphasis added).

Claim 10

Claim 10 recites:

The process of claim 1 where the methods in one or more classes are grouped in a
package where the package includes a package data structure including first and
second portions, the process including storing the exception handler table in the first
portion of the package and all methods in the second portion of the package.

The Examiner states:

As to claim 10, Yellin further teaches the methods in one or more classes are grouped in a
package (Java class file, line 20 column 3) where the package includes a package data
structure including first and second portions (Fig. 2), the process including storing the
exception handler table in the first portion (ThreadDeath to NoSuchMethod Error, Fig. 2)
of the package and all methods in the second portion (thrownable, error and exception,
Fig. 122) of the package.[19]

The Applicants respectfully disagree. Contrary to the Examiner's statement, Yellin et al.

does not disclose where the methods in one or more classes are grouped in a package where the

package includes a package data structure including first and second portions, the process

including storing the exception handler table in the first portion of the package and all methods

in the second portion of the package. In support of the contention that Yellin et al. discloses

methods in one or more classes grouped in a package, the Examiner refers to a Java™ class file.

The Applicants submit that equating a class file with a package is improper. As mentioned in the

Specification, one or more classes can be grouped in a package where the package includes a

package data structure including first portion including an exception handler table, and a second

portion including all the methods.[20] Since this element is not disclosed in the Yellin et al.

reference, the rejection is unsupported by the art and should be withdrawn.

---

[18] Office Action at ¶ 4.
[19] Office Action at ¶ 4.

Claim 11

Claim 11 recites:

The process of claim 10 where the step of combining includes concatenating the
exception handler arrays including loading each exception handler array into the first
portion of the package data structure in accordance with a predefined ordering.

The Examiner states:

As to claim 11, Yellin further teaches concatenating the exception handler arrays
(hierarchy shown in Figs. 2-3) including loading each exception handler array into the
first portion (ThreadDeath to NoSuchMethodError, Fig. 2) of the package data structure
in accordance with a predefined ordering (class rank, line 49 column 3).[21]

The Applicants respectfully disagree. Contrary to the Examiner's statement, Yellin et al.

does not disclose concatenating the exception handler arrays including loading each exception

handler array into the first portion of the package data structure in accordance with a predefined

ordering. In support of the contention that Yellin et al. discloses concatenating exception handler

arrays, the Examiner refers to block diagrams showing portions of an exception class hierarchy.

The Applicants submit that equating a hierarchical relationship between exceptions with

concatenating exception handler *arrays* is improper. Since this element is not disclosed in the

Yellin et al. reference, the rejection is unsupported by the art and should be withdrawn.

Claim 27

Claim 27 as amended recites:

A computer implemented system for managing exceptions throwable during execution of
methods in one or more classes on a resource-constrained device, each method
including an exception handler array defining exception handlers associated with the
method, the system comprising instructions to:

---

[20] Specification at p. 9 lines 6-12.
[21] Office Action at ¶ 4.
[22] *Id.*

30

combine the exception handler arrays for all methods into a single exception handler table.

The Examiner states:

As to the system of claim 27, note the discussions of claims 1 and 2 above.[23]

Claim 27 includes limitations similar to claims 1 and 2. Thus, the arguments made with respect to claims 1 and 2 apply here as well. Claims 1 and 2 being allowable, claim 27 must also be allowable.

In view of the foregoing, it is respectfully asserted that the claims are now in condition for allowance. It is respectfully requested that the 35 U.S.C. § 102 rejection of claims based on Yellin et al. be withdrawn.

The First 35 U.S.C. § 103 Rejection

Claims 13-16, 25-26 and 29-30 stand rejected under 35 U.S.C. § 103(a) as being allegedly unpatentable over Yellin et al. in view of Levy et al.[24][25] This rejection is respectfully traversed.

According to the M.P.E.P.,

To establish a *prima facie* case of obviousness, three basic criteria must be met. First there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine reference teachings. Second, there must be a reasonable expectation of success. Finally, the prior art reference (or references when combined) must teach or suggest all the claim limitations. The teaching or suggestion to make the claimed

---

[23] *Id.*
[24] U.S. Patent No. 6,092,147.
[25] Office Action at ¶ 5.

combination and the reasonable expectation of success must both be found in the prior art, not in the applicant's disclosure.[26]

Claim 13

Claim 13 as amended recites:

The process of claim 1 where the resource-constrained device comprises a virtual machine configured to perform said execution of said methods.

The Examiner states:

As to claim 13, Yellin further discloses the machine is a virtual machine (lines 11 57 column 3). However, Yellin does not teach a resource-constrained device. Levy teaches implementing a virtual machine on a resource-constrained device (Fig. 1). It would have been obvious to apply the teachings of Levy to the system of Yellin because this provides a reduction in the overall memory size and an increase in the overall processing speed of the virtual machine as disclosed by Levy (lines 33-55 column 2).[28]

The Applicants respectfully disagree for the reasons set forth below.

I. Yellin et al. and Levy et al. Do Not Teach All Claim Limitations

When evaluating a claim for determining obviousness, all limitations of the claim must be evaluated.[29]

Claim 13 depends from claim 1. As noted above, Yellin et al. does not anticipate claim 1 because Yellin et al. does not teach or suggest all claim limitations. For the same reasons, Yellin et al. and Levy et al. cannot be said to make claim 13 obvious. Thus, the Examiner has failed to make a prima facie case of obviousness so the 35 U.S.C. § 103 rejection should be withdrawn.

---

[26] Manual of Patent Examining Procedure (M.P.E.P) § 2143.
[27] In re Mills, 916 F.2d 680, 16USPQ2d 1430 (Fed. Cir. 1990).
[28] Office Action at ¶ 5.

II. There Is No Basis in the Art for Combining or Modifying Yellin et al. and Levy et al.

MPEP § 2143 provides:

> The mere fact that references *can* be combined or modified does not render the resultant combination obvious unless the prior art also suggests the desirability of the combination.[30] (emphasis added)

Furthermore,

> Obviousness cannot be established by combining the teachings of the prior art to produce the claimed invention, absent some teaching, suggestion or incentive supporting the combination.[31]

The Applicants submit there is no basis for combining or modifying Yellin et al. and Levy et al. The Examiner has stated the motivation to combine the references is that it would result in a reduction in the overall memory size and an increase in the overall processing speed of the virtual machine.[32] However, what the Examiner has attempted to construe as a motivation to combine references is in fact a motivation to use a single reference. To suggest otherwise would constitute hindsight reconstruction. The Federal Circuit has stated:

> To prevent the use of hindsight based on the invention to defeat patentability of the invention, this court requires the examiner to show a motivation to combine the references that create the case of obviousness. In other words, the examiner must show reasons that the skilled artisan, confronted with the same problems as the inventor and with no knowledge of the claimed invention, would select the elements from the cited prior art references for combination in the manner claimed.[33]

---

[29] *In re Dillon*, 919 F.2d 688, 16 USPQ2d 1897 (Fed. Cir. 1990).
[30] *In re Mills*, 916 F.2d 680, 16 USPQ2d 1430 (Fed. Cir. 1990).
[31] *ACS Hospital Systems, Inc. v. Monteffore Hospital*, 732 F.2d 1572, 1577, 221 USPQ 929, 933 (Fed. Cir. 1984).
[32] Office Action at ¶ 5.
[33] *In re Rouffet*, 149 F.3d 1350, 47 USPQ2d 1453 (Fed. Cir. 1998).

As stated in the Specification, the problems the inventor faced involved limited memory capacity on resource constrained devices[34]. The reduction in overall memory size and increase in the overall processing speed of the virtual machine are addressed by <u>Levy et al.</u> alone. The Examiner has not shown why a skilled artisan would go a step further in combining <u>Levy et al.</u> with <u>Yellin et al.</u> Nor does the suggestion to combine come from <u>Yellin et al.</u> In fact, <u>Yellin et al.</u> indicates it makes *no difference* whether there is a separate table of exception handlers for each method, or just one for all the methods in a class.[35]

Additionally, the Applicants submit the reasons for the combination of references suggested by the Examiner do not constitute particular findings as required by the Federal Circuit. The Federal Circuit has stated:

> The motivation, suggestion or teaching may come explicitly from statements in the prior art, the knowledge of one of ordinary skill in the art, or, in some cases the nature of the problem to be solved. In addition, the teaching, motivation or suggestion may be implicit from the prior art as a whole, rather than expressly stated in the references. ... The test for an implicit showing is what the combined teachings, knowledge of one of ordinary skill in the art, and the nature of the problem to be solved as a whole would have suggested to those of ordinary skill in the art. ... Whether the Board relies on an express or an implicit showing, it must provide particular findings related thereto. Broad conclusory statements standing alone are not "evidence."[36]

The Examiner's broad conclusory statement that "It would have been obvious to apply the teachings of Levy to the system of Yellin because this provides a reduction in the overall memory size and an increase in the overall processing speed of the virtual machine as disclosed by Levy (lines 33-55 column 2)."[37] standing alone is not evidence.

---

[34] <u>Specification</u> at p. 7 line 33 to p. 8 line 2.
[35] <u>Yellin et al.</u> at col. 3 lines 16-18. (emphasis added)
[36] *In re Kotzab*, 217 F.3d 1370, 55 USPQ2d 1317 (Fed. Cir. 2000) (citations omitted).
[37] Office Action at ¶ 5.

For this additional reason, the Examiner has failed to make a *prima facie* case of obviousness so the 35 U.S.C. § 103 rejection should be withdrawn.

III. Yellin et al. and Levy et al. Are Not Properly Combinable or Modifiable Because Such a Combination Would Destroy Their Intended Function.

The CCPA had and the Federal Circuit have consistently held that when a § 103 rejection is based upon a modification of a reference that destroys the intent, purpose or function of the invention disclosed in the reference, such a proposed modification is not proper and the *prima facie* case of obviousness cannot be properly made.[38]

The Yellin et al. reference requires *additional* code be executed upon initiating execution of a method to inspect the method's declaration of throwable exceptions.[39] The modification proposed in the Examiner's rejection, i.e., implementing the system of Yellin et al. on a resource-constrained device, would render both Yellin et al. and Levy et al. inoperable for their intended purposes. The so-modified Yellin. et al. construction would lose its method declaration inspection feature, and the modified Levy et al. construction would be inoperable for executing on a resource-constrained device due to the additional code required to inspect method declarations.

For this additional reason, the Examiner has failed to make a *prima facie* case of obviousness so the 35 U.S.C. § 103 rejection should be withdrawn.

---

[38] See e.g., *In re Gordon*, 733 F.2d 900, 221 USPQ 1125 (Fed. Cir. 1984).
[39] Yellin et al. Abstract. (emphasis added)

IV. Yellin et al. and Levy et al. Are Not Properly Combinable or Modifiable Because The Prior Art Teaches Away From the Claimed Invention.

The Federal Circuit has held that teaching away is the antithesis of the art suggesting that the person of ordinary skill in the art go in the claimed direction.[40] Only when the PTO makes a prima facie case of obviousness does the burden of coming forward shift to the applicant.[41]

The Yellin et al. reference indicates it is preferable to implement Yellin et al.'s invention on client computer configured with a plethora of features. Yellin et al. states:

> A workstation or other computer 200 implementing the preferred embodiment of the present invention will typically be a client computer in a computer network that includes other client computers 200 as well as one or more servers 202, all interconnected via one or more communications media 203.
> More specifically, computer workstation 200 includes a central processing unit 204, a user interface 206, a communications interface 207 such as an Internet communications port, and memory 208.
> Memory stores:
> an operating system 210;
> an Internet communications manager program 212, such as the HotJava (a trademark of Sun Microsystems, Inc.) Web browser program;
> a Java bytecode program verifier 214 for verifying whether or not a specified program (i.e. method) satisfies certain predefined integrity criteria;
> a Java bytecode program interpreter 216 for executing Java bytecode programs loaded into an interpreter work array 218;
> Java class reslover 220, which dynamically links methods during execution of those methods, including determining if an invoked method is already loaded and calling the class loader 222 if the invoked method is not already loaded;
> a Java class loader 222, which loads object classes into a user's address space and utilizes the bytecode program verifier to verify the integrity of the methods associated with each loaded object class;
> a Java class table 224, which references all loaded object classes;
> a compiler 226 for compiling Java source code programs into Java bytecode programs;
> at least one repository 228, for locally storing object classes (i.e., class files) 230 in use and/or available for use by user's of the computer 200;
> at least one object repository 232 for storing objects 234, which are instances of objects of the object classes stored in the object repository.[42]

---

[40] *In re Fine*, 873 F.2d 1071, 5 USPQ 2d 1596 (Fed Cir. 1988).
[41] *In re Hedges*, 783 F.2d 1038, 228 USPQ 685 (Fed. Cir. 1986).

Yellin et al. further states in the preferred embodiment, the operating system is an object-oriented multitasking operating system that supports multiple threads of execution within each defined address space.[43]

As mentioned in the Specification, resource-constrained devices are generally considered to be those that are restricted in memory and/or computing power or speed[44] and thus ill-suited for implementing aspects of Yellin et al.

Thus, Yellin et al. teaches away from using a resource constrained device. For this additional reason, the Examiner has failed to make a *prima facie* case of obviousness so the 35 U.S.C. § 103 rejection should be withdrawn.

Dependent Claims 14-16

Claims 14-16 depend from claim 13. The base claim being allowable, the dependent claims must also be allowable.

Claim 15

Claim 15 recites:

The process of claim 14 where the methods in one or more classes are grouped in a package and the package is installed on the smart card.

The Examiner states:

---

[42] Yellin et al. at col. 4 line 44 to col. 5 line 18.
[43] *Id.* at col. 5 lines 18-22.
[44] Specification at p. 14 lines 18-21.
[45] Office Action at ¶ 5.

As to claim 15, note the discussion of claim 10.[46]

The arguments made above with respect to claim 10 apply here. Contrary to the Examiner's statement, <u>Yellin et al.</u> does not disclose where the methods in one or more classes are grouped in a package and the package is installed on the smart card.

Claim 16

Claim 16 recites:

> The process of claim 15 further including creating a package where the package includes a package data structure including first and second portions, the process including concatenating the exception handler arrays for each of the methods into a exception handler table, storing the exception handler table in the first portion of the package and all methods in the second portion of the package.

The Examiner states:

As to claim 16, note the discussions of claims 2 and 10.[47]

The arguments made above with respect to claim 10 apply here. Contrary to the Examiner's statement, <u>Yellin et al.</u> does not disclose creating a package where the package includes a package data structure including first and second portions, the process including concatenating the exception handler arrays for each of the methods into a exception handler table, storing the exception handler table in the first portion of the package and all methods in the second portion of the package.

Claim 25

---

[46] *Id.*

Claim 25 recites:

> A method of converting class files into a converted applet for execution on a resource
> constrained device including;
>> receiving one or more class files, each class file including one or more methods, each
>> method including an exception handler array defining exception handlers
>> catchable by the method;
>> defining a data structure for storing the methods and exception handlers for the
>> converted applet including a first and second portion;
>> defining an ordering for the methods and loading the methods according to the
>> ordering in the second portion of the data structure; and
>> combining the exception handler arrays for all methods in a single exception handler
>> table including ordering the exception handler arrays according to the ordering
>> defined for the methods and storing the single exception handler array in the
>> first portion of the data structure.

The Examiner states:

> As to the method of claim 25, note the discussions of claims 1 and 10-12 above.[48]

Claim 25 includes limitations similar to those of claims 1 and 10-12. Thus, the

arguments made above with respect to claim 1 and 10-12 apply here as well. Claims 1 and 10-12

being allowable, claim 25 must also be allowable.

Claim 26

Claim 26 as amended recites:

> A computer implemented process for managing exceptions throwable during execution of
> two or more methods in one or more classes by a virtual machine on a resource-
> constrained device, each method included in a class and including an exception
> handler array defining exception handlers associated with the method, the individual
> exception handler arrays combined and forming a single exception handler table for
> the two or more methods, the process comprising:
>> searching the exception handler table when an exception is thrown while executing
>> one of the methods including locating a first matching exception in the single
>> exception handler table.

---

[47] *Id.*
[48] *Id.*

The Examiner states:

As to the process of claim 26, note the discussions of claims 1, 6 and 13 above.[49]

Claim 26 includes limitations similar to those of claims 1 and 6 and 13. Thus, the arguments made above with respect to claim 1, 6 and 13 apply here as well. Claims 1, 6 and 13 being allowable, claim 26 must also be allowable.

Claim 29

Claim 29 as amended recites:

A computer implemented system for converting class files into a converted applet for
    execution on a resource constrained device, the system comprising instructions to:
    receive one or more class files, each class file including one or more methods, each
        method including an exception handler array defining exception handlers
        catchable by the method;
    define a data structure for storing the methods and exception handlers for the
        converted applet including a first and second portion;
    define an ordering for the methods and loading the methods according to the
        ordering in the second portion of the data structure; and
    combine the exception handler arrays for all methods in a single exception handler
        table including order the exception handler arrays according to the order defined
        for the methods and store the single exception handler array in the first portion
        of the data structure.

The Examiner states:

As to the system of claim 29, note the discussions of claims 1 and 10-12 above.[50]

---

[49] *Id.*
[50] *Id.*

Claim 29 includes limitations similar to those of claims 1 and 10-12. Thus, the

arguments made above with respect to claim 1 and 10-12 apply here as well. Claims 1 and 10-12

being allowable, claim 29 must also be allowable.

Claim 30

Claim 30 as amended recites:

A computer implemented system for managing exceptions throwable during execution of
methods in one or more classes by a virtual machine on a resource-constrained
device, each method in a class described by a class file and including an exception
handler array defining exception handlers associated with the method, the individual
exception handler arrays combined and forming a single exception handler table for
two or more methods, the system comprising instructions to:
search the exception handler table when an exception is thrown while executing one of
the two or more methods including locate a first matching exception in the single
exception handler table.

The Examiner states:

As to the system of claim 30, note the discussions of claims 1, 5-6 and 13 above.[51]

Claim 30 includes limitations similar to those of claims 1, 5-6 and 13. Thus, the

arguments made above with respect to claim 1, 5-6 and 13 apply here as well. Claims 1, 5-6 and

13 being allowable, claim 30 must also be allowable.

In view of the foregoing, it is respectfully asserted that the claims are now in condition

for allowance. It is respectfully requested that the 35 U.S.C. § 103 rejection of claims based on

Yellin et al. in view of Levy et al. be withdrawn.

---

[51] *Id.*

41

The Second 35 U.S.C. § 103 Rejection

Claims 17-19 and 28 stand rejected under 35 U.S.C. 103(a) as being unpatentable over

Yellin et al. in view of Bak et al.[52] [53] This rejection is respectfully traversed.

Claim 17

Claim 17 as amended recites:

A method minimizing the amount of storage required for a runtime stack when executing
    a program, the runtime stack maintained at runtime during the execution of the
    program on a resource-constrained device for storing one or more frames where each
    frame includes a return pointer to an invoking method that called a currently
    executing method in the program, the method comprising:
combining exception handler information for methods included in the program into a
    combined exception handler table; and
locating and searching the combined exception handler table when an exception is
    thrown during execution of one of the methods to locate the exception handler
    information without requiring the storage on the runtime stack of a pointer to the
    exception handler information.

The Examiner states:

As to claim 17, note the discussions of claims 1 and 6 above. However, Yellin as
modified does not discloses a return pointer. Bak discloses a stack with frames wherein
each frame includes a return pointer (line 52 column 2 to line 41 column 3). It would
have been obvious to apply the teachings of Bak to the system of Yellin as modified
because this allows exceptions to propagate through the execution stack for handling by
the appropriate exception handler, even when the functions were written in different
languages and the format of the exceptions are different as disclosed by Bak (lines 52
column 2 to line 7 column 3).[54]

---

[52] U.S. Patent No. 6,009,517.
[53] Office Action at ¶ 6.
[54] *Id.*

With this Amendment, claim 17 has been modified to specify execution on a resource-constrained device. The resource-constrained device element is disclosed in neither the <u>Yellin et al.</u> reference nor the <u>Bak et al.</u> reference, so the rejection is unsupported by the art and should be withdrawn.

Dependent Claims 18-19

Claims 18-19 depend from claim 17. The base claim being allowable, the dependent claims must also be allowable.

Claim 28

Claim 28 as amended recites:

> A computer implemented system for minimizing the amount of storage required for a runtime stack when executing a program, the runtime stack maintained at runtime during the execution of the program on a resource-constrained device for storing one or more frames where each frame includes a return pointer to an invoking method that called a currently executing method in the program, the system comprising instructions to:
> combine the exception handler information for two or more methods included in the program into a combined exception handler table; and
> locate and search the combined exception handler table when an exception is thrown during execution of one of the methods to locate the exception handler information without requiring the storage on the runtime stack of a pointer to the exception handler information.

The Examiner states:

As to the system of claim 28, note the discussion of claim 17 above.[57]

---

[55] *Id.*
[56] *Id.*
[57] *Id.*

Claim 28 includes limitations similar to those of claim 17. Thus, the arguments made above with respect to claim 17 apply here as well. Claim 17 being allowable, claim 28 must also be allowable.

In view of the foregoing, it is respectfully asserted that the claims are now in condition for allowance. It is respectfully requested that the 35 U.S.C. § 103 rejection of claims based on Yellin et al. in view of Bak et al. be withdrawn.

The Third 35 U.S.C. § 103 Rejection

Claims 20-24 stand rejected under 35 U.S.C. 103(a) as being unpatentable over Yellin et al. in view of Levy et al., and further in view of Bak et al.[58] This rejection is respectfully traversed.

Claim 20

Claim 20 as amended recites:

> The method of claim 19 where the resource-constrained device comprises a virtual machine implementing a Java virtual machine and configured to perform said execution of said methods.

The Examiner states:

> As to claim 20, note the discussion of claim 13 above.[59]

---

[58] Office Action at ¶ 7.
[59] *Id.*

Claim 20 includes limitations similar to those of claim 13. Thus, the arguments made above with respect to claim 13 apply here as well. Claim 13 being allowable, claim 20 must also be allowable.

Dependent Claims 21-24

Claims 21-24 depend from claim 20. The base claim being allowable, the dependent claims must also be allowable.

Claim 21

Claim 21 as amended recites:

The method of claim 20 where the program includes a package of methods, the methods in one or more classes, and where the virtual machine is implemented in a resource constrained device on which the package is installed and executing.

The Examiner states:

As to claim 21, note the discussions of claims 2 and 13 above.[60]

Claim 21 includes limitations similar to those of claims 2 and 13. Thus, the arguments made above with respect to claim 2 and 13 apply here as well. Claims 2 and 13 being allowable, claim 21 must also be allowable.

Claim 23

Claim 23 recites:

---

[60] *Id.*
[61] *Id.*

The method of claim 21 further including registering the package in a registry service at installation, the registry service maintaining a pointer and a range, the pointer indicating a location in the resource constrained device of the combined exception handler table associated with a given package, the range defining a range of addresses in the resource constrained device at which methods associated with the package are located.

The Examiner states:

As to claim 23, Yellin as modified further discloses registering the package in a registry service at installation (line 11 column 3 to line 9 column 4), the registry service maintaining a pointer indicating a location of the combined exception handler table (lines 41-57 column 3), and a range defining a range of addresses at which methods are located (lines 9-18 column 3).[62]

The Applicants respectfully disagree. Contrary to the Examiner's statement, the cited reference does not disclose a registry service or an installation process. Furthermore, the reference does not disclose a pointer indicating a location in the resource constrained device of the combined exception handler table. Further still, the cited reference does not disclose a package structure, as mentioned previously with regard to claim 10.

Claim 24

Claim 24 recites:

The method of claim 23 where the step of locating includes locating a package associated with a currently executing method including comparing an address at which an exception was thrown against the range for each package registered in the registry service, the searching step including searching the combined exception handler table associated with a located package.

The Examiner states:

As to claim 24, Yellin as modified further discloses locating a package associated with a currently executing method including comparing an address at which an exception was thrown against the range for each package registered in the registry service (line 58

---

[62] *Id.*

column 3 to line 9 column 4), searching the combined exception handler table associated with a located package (lines 41-57 column 3).[63]

The Applicants respectfully disagree. The arguments made with respect to claim 10 apply here. Contrary to the Examiner's statement, the Yellin et al. reference does not disclose a package structure.

In view of the foregoing, it is respectfully asserted that the claims are now in condition for allowance. It is respectfully requested that the 35 U.S.C. § 103 rejection of claims based on Yellin et al. in view of Levy et al. and further in view of Bak et al. be withdrawn.

The attached page is captioned "**Version with markings to show changes made.**"

In view of the foregoing, it is respectfully asserted that the claims are now in condition for allowance.

---

[63] *Id.*

Request for Allowance

It is believed that this Response places the above-identified patent application into condition for allowance. Early favorable consideration of this application is earnestly solicited.

If, in the opinion of the Examiner, an interview would expedite the prosecution of this application, the Examiner is invited to call the undersigned attorney at the number indicated below.

Respectfully submitted,
THELEN REID & PRIEST, LLP

Dated: January 30, 2003

John P. Schaub
Reg. No. 42,125

THELEN REID & PRIEST LLP
P. O. Box 640640
San Jose, CA 95164-0640
Tel: (408) 292-5800
#120998V1

Version With Markings To Show Changes Made

1. (Amended Once) A computer implemented process for managing exceptions throwable

during execution of methods in one or more classes <u>on a resource-constrained device</u>, each

method including an exception handler array defining exception handlers associated with the

method, the process comprising:

combining the exception handler arrays for two or more methods into a single exception

handler table.

13. (Amended Once) The process of claim 1 where the [machine] <u>resource-constrained device</u>

[is] <u>comprises</u> a virtual machine [implemented on a] <u>configured to perform said execution</u>

<u>of said methods</u> [resource constrained device].

17. (Amended Once) A method minimizing the amount of storage required for a runtime stack

when executing a program, the runtime stack maintained at runtime during the execution of

the program [by a machine] <u>on a resource-constrained device</u> for storing one or more frames

where each frame includes a return pointer to an invoking method that called a currently

executing method in the program, the method comprising:

combining exception handler information for methods included in the program into a

combined exception handler table; and

locating and searching the combined exception handler table when an exception is thrown

during execution of one of the methods to locate the exception handler information

without requiring the storage on the runtime stack of a pointer to the exception handler

information.

49

20. (Amended Once) The method of claim 19 where the [machine] <u>resource-constrained device</u> [is] <u>comprises</u> a virtual machine implementing a Java virtual machine <u>and configured to perform said execution of said methods</u>.

21. (Amended Once) The method of claim 20 where the program includes a package of methods, the methods in one or more classes, and where the virtual machine is implemented in a resource constrained device on which the package is installed and executing.

26. (Amended Once) A computer implemented process for managing exceptions throwable during execution of two or more methods in one or more classes by a virtual machine <u>on a resource-constrained device</u>, each method included in a class and including an exception handler array defining exception handlers associated with the method, the individual exception handler arrays combined and forming a single exception handler table for the two or more methods, the process comprising:

searching the exception handler table when an exception is thrown while executing one of

the methods including locating a first matching exception in the single exception

handler table.

27. (Amended Once) A computer implemented system for managing exceptions throwable during execution of methods in one or more classes [by a machine] <u>on a resource-constrained device</u>, each method including an exception handler array defining exception handlers associated with the method, the system comprising instructions to:

combine the exception handler arrays for all methods into a single exception handler table.

28. (Amended Once) A computer implemented system for minimizing the amount of storage required for a runtime stack when executing a program, the runtime stack maintained at runtime during the execution of the program [by a machine] on a resource-constrained device for storing one or more frames where each frame includes a return pointer to an invoking method that called a currently executing method in the program, the system comprising instructions to:

combine the exception handler information for two or more methods included in the program into a combined exception handler table; and

locate and search the combined exception handler table when an exception is thrown during execution of one of the methods to locate the exception handler information without requiring the storage on the runtime stack of a pointer to the exception handler information.

29. (Amended Once) A computer implemented system for converting class files into a converted applet for execution on a resource constrained device, the system comprising instructions to:

receive one or more class files, each class file including one or more methods, each method including an exception handler array defining exception handlers catchable by the method;

define a data structure for storing the methods and exception handlers for the converted applet including a first and second portion;

51

define an ordering for the methods and loading the methods according to the ordering in the

second portion of the data structure; and

combine the exception handler arrays for all methods in a single exception handler table

including order the exception handler arrays according to the order defined for the

methods and store the single exception handler array in the first portion of the data

structure.

30. (Amended Once) A computer implemented system for managing exceptions throwable

during execution of methods in one or more classes by a virtual machine on a resource-

constrained device, each method in a class described by a class file and including an

exception handler array defining exception handlers associated with the method, the

individual exception handler arrays combined and forming a single exception handler table

for two or more methods, the system comprising instructions to:

search the exception handler table when an exception is thrown while executing one of the

two or more methods including locate a first matching exception in the single exception

handler table.